Alibaba Cloud ApsaraVideo for Media Processing

開發指南

檔案版本：20180929

目錄

1 概念介紹

1.1 作業和管道

本文介紹ApsaraVideo for Media Processing服務的幾個基本概念和關係，以便您更好的理解和使用ApsaraVideo for Media Processing服務。

示意圖：

概念解釋：

- 作業

  作業在ApsaraVideo for Media Processing服務裡面是一個抽象的概念，包含多種類型的作業：轉碼作業、截圖作業、媒體資訊作業等。

  在一個作業中，包含3個關鍵資訊：輸入、輸出和參數。輸入和輸出設定作業執行時的輸入檔案以及執行後的輸出檔案，參數則用來設定執行具體功能的詳細配置。

- 參數

  — 模板參數

  由於作業的參數很多，每次提交作業時都重複填寫比較麻煩，模板是為解決此問題而提出的概念。模板是一些參數的合集，把一些常用的參數組合在一起，可以減少提交作業的參數數量，簡化提交作業的代碼。

  — API參數

如果給每種不同的參數組合都建立模板，會導致模板的數量劇增，也使得範本管理員變的複雜。所以不僅可以在模板中設定參數，也可以在實際調用API時設定對應的參數。

- 覆蓋順序

  API參數比模板中對應參數的優先順序更高，會覆蓋模板中對應的參數。

  舉個例子：同一個視頻可以轉碼輸出多種清晰度（高清、標清等），不同清晰度的容器格式（MP4)、編碼通訊協定(H.264)、幀率（25幀）是相同的，區別只是碼率和解析度的不同。就可以先建立（MP4+H.264+25FPS+2Mbps+1280x720）這樣一個預設參數組合模板。調用API時，如果不設定API參數，則按照預設參數（2Mbps+1280x720）執行作業，如果設定API參數（4Mbps+1920x1080），這按照API參數（4Mbps+1920x1080）執行作業。

- 管道

  當使用者通過API介面提交作業後，作業會先進入管道中進行排隊，根據優先順序和提交順序依次執行。

  管道中的作業可以有多種優先順序（最高10，最低1，預設6），相同優先順序的作業之間，提交作業時間早的比晚的先執行，不同優先順序的作業之間，高優先順序的比低優先順序的先執行。

- 作業執行和結果

  - 同步和非同步

    根據作業的類型不同，一些作業能很快完成，但是大部分的作業都無法即時完成，作業執行有2種方式：同步和非同步。同步方式（例如，截圖作業）會立即返回結果，非同步方式（例如，轉碼作業）的結果有2種查詢方式：定時輪詢和訊息通知。

  - 定時輪詢

    每個作業都有一個唯一作業ID來標識，在提交作業時會同步返回給調用方，之後可以通過作業ID查詢作業結果。這種方式的缺點是不夠即時。

    訊息通知：

    管道可以配置訊息通知，就能及時獲得作業結果。訊息通知中包含了幾個重要的資訊：作業ID、使用者資料和結果。

    - 作業ID

      提交作業時記錄下作業ID，然後和訊息通知的作業ID對比，就能知道是哪個作業的結果。

    - 使用者資料

提交作業時，可以填寫自訂的使用者資料參數（例如，商品ID），然後訊息通知中會返回自訂的使用者資料參數，這樣可以不需要在業務系統中記錄作業ID，使用自訂的使用者資料（例如，商品ID）來關聯業務系統。

1.2 轉碼模板

由於轉碼作業的參數很多，每次提交轉碼作業時都重複填寫比較麻煩，轉碼模板是為解決此問題而提出的概念，本質就是把一些常用的參數組合在一起。轉碼模板提供了2種類型：預置模板和自訂模板。

- 預置模板

  根據常見的一些參數組合，預先提供給使用者的轉碼模板。參見 ######。

  預置模板又包括幾個細分類型：

  - 預置靜態模板

    可以直接使用的轉碼模板，包含視頻轉碼、音頻轉碼、轉封裝等各種情境。例如，“MP4-高清”、“MP3-128K”。

  - 窄帶高清預置模板

    窄帶高清是媒體轉碼特有的技術。在相同的碼率下，能帶來更高的清晰度，這樣可以在相同成本下提供更好的使用者體驗。

  - 預置智能模板

    預置智能模板能自動根據輸入檔案內容的特點動態調整轉碼參數，在相同的清晰度下，能帶來更低的碼率，這樣可以節省更多的成本。

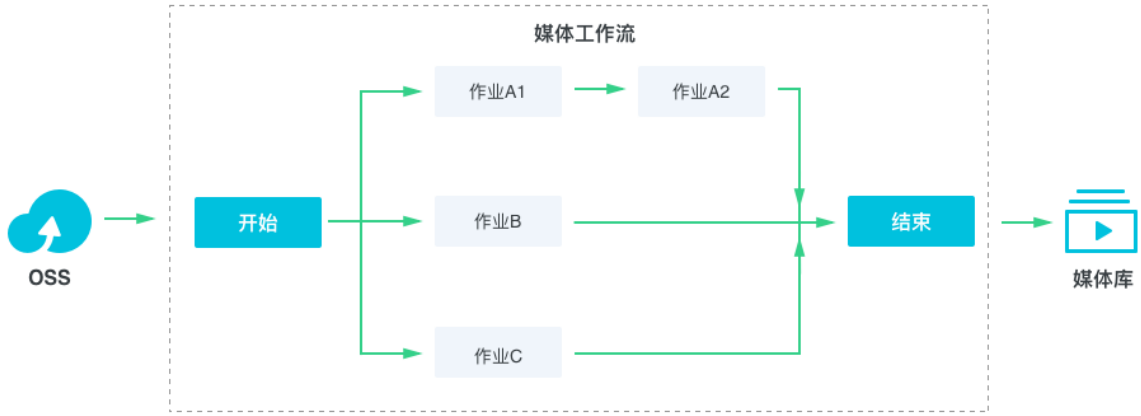
    > 说明：
    > 使用預置智能模板時，要先調用 提交模板分析作業 介面（**SubmitAnalysisJob**），分析作業成功完成後，可以調用 查詢範本分析作業 介面（**QueryAnalysisJobList**）擷取該輸入檔案對應的有效預置智能模板列表。若提交的轉碼作業中指定的預置智能模板不在有效列表中，則轉碼作業是無效的，會返回失敗。

- 自訂模板

  預置的模板不能滿足需求時，可以使用自訂模板來自行定義轉碼參數(音頻、視頻、容器、轉碼等)的組合。每個自訂模板有一個唯一模板ID。

1.3 工作流程和媒體

本文介紹ApsaraVideo for Media Processing服務的幾個基本概念和關係，以便您更好的理解和使用ApsaraVideo for Media Processing服務。



概念解釋：

- 媒體

  媒體包含一個輸入（視頻、音頻多媒體檔案）和相關的所有輸出（例如，轉碼、截圖、媒體資訊、AI標籤等）。輸入和媒體是一一對應的，由媒體ID唯一標識。

  媒體庫

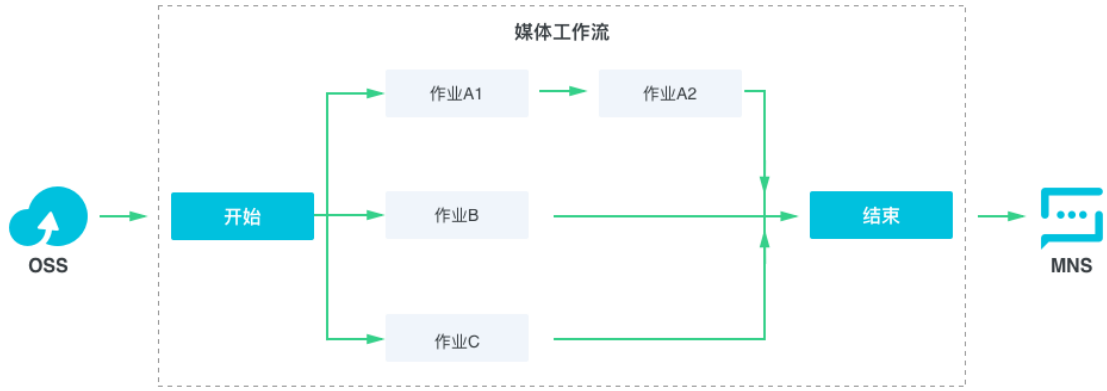
  媒體庫是所有媒體的集合，媒體是媒體庫的最小嵌入式管理單元。

- 媒體工作流程

  媒體工作流程是自動化生產媒體的工廠，由MediaWorkflowId唯一標識。

  > 说明：
  > 媒體工作流程在文檔中也簡稱為工作流程。

  - 活動



工作流程中的每個節點稱為活動。根據實際需求，即可以並存執行（例如，示意圖的作業A、B、C之間），也可以串列執行（例如：示意圖的作業A1、A2之間）。除開始的輸入活動和結束的發布彙報活動，活動支援各種類型的作業（轉碼作業、截圖作業等）。

- 開始的輸入活動

  配置工作流程關聯儲存的觸發路徑，只要在對應的路徑上傳視頻、音頻多媒體檔案，就會自動觸發工作執行。

- 結束的發布彙報活動

  工作流程執行完成後，會訊息通知執行結果。執行結果包含了媒體ID和多媒體檔案的絕對位址，這樣就能對應具體是哪個多媒體檔案執行完成。

- 作業活動

  作業支援的所有參數，都可以在作業活動中配置。

- 路徑匹配規則

  匹配使用路徑首碼的規則，例如，上傳的檔案是 `http://bucket.oss-cn-hangzhou.aliyuncs.com/A/B/C/test1.flv`，配置的觸發路徑結果如下：

路徑	是否匹配
http://bucket.oss-cn-hangzhou.aliyuncs.com/A/B/C/	是
http://bucket.oss-cn-hangzhou.aliyuncs.com/A/B/C2/	否

路徑	是否匹配
http://bucket.oss-cn-hangzhou.aliyuncs.com/A/B/	是
http://bucket.oss-cn-hangzhou.aliyuncs.com/A/B2/	否
http://bucket.oss-cn-hangzhou.aliyuncs.com/A/	是
http://bucket.oss-cn-hangzhou.aliyuncs.com/A2/B/C/	否
http://bucket.oss-cn-hangzhou.aliyuncs.com/A/B/C/test	是
http://bucket.oss-cn-hangzhou.aliyuncs.com/A/B/C/test2	否

- 副檔名匹配規則

  上傳時的自動觸發機制會檢查檔案的副檔名，避免產生一些無效的資料（例如pdf、word文檔等）。

   说明：
  API手動觸發機制不檢查副檔名。

  檔案或者沒有副檔名(檔案名稱中不包含副檔名分割符號”.”)，或者副檔名符合下面的規則：

  - 視頻

    3gp、asf、avi、dat、dv、flv、f4v、gif、m2t、m3u8、m4v、mj2、mjpeg、mkv、mov、mp4、mpe、mpg、mpeg、mts、ogg、qt、rm、rmvb、swf、ts、vob、wmv、webm

  - 音頻

    aac、ac3、acm、amr、ape、caf、flac、m4a、mp3、ra、wav、wma、aiff

- 工作流程執行

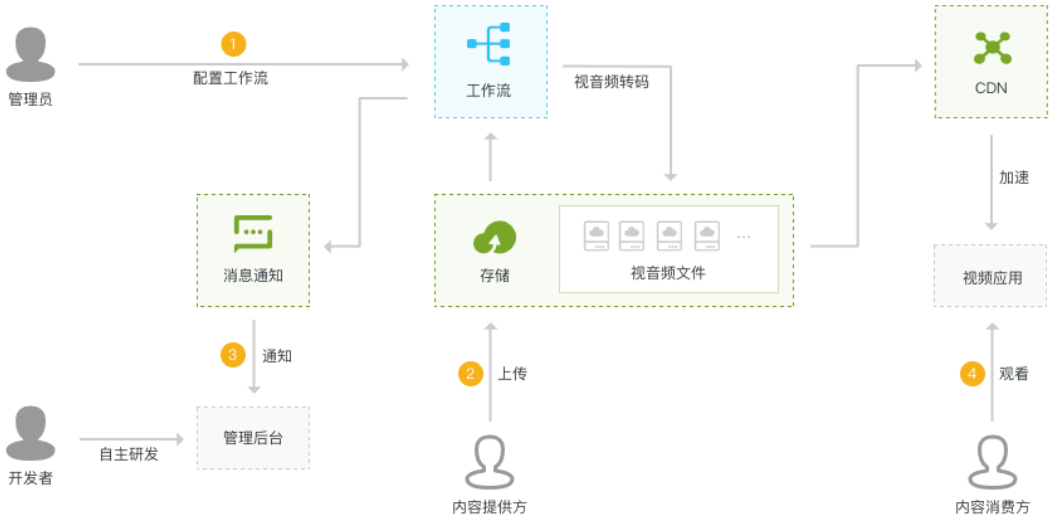
  每次上傳匹配的多媒體檔案都會觸發一次執行，同一個多媒體檔案如果多次上傳，則會觸發多次執行，每次執行有唯一的RunId標識。

  除了上傳時的自動觸發機制，工作流程針對儲存中的存量多媒體檔案，也提供了API手動觸發機制。每次調用API都會觸發一次執行。

- 使用者資料

每次執行時，可以填寫自訂的使用者資料參數（例如，商品ID），然後訊息通知中會返回自訂的使用者資料參數，這樣可以不需要在業務系統中記錄媒體ID或者多媒體檔案的絕對路徑，使用自訂的使用者資料（例如，商品ID）來關聯業務系統。

2 工作流程的開發流程



1. 配置媒體工作流程。

   簡單易用：通過控制台的圖形化介面，按需搭建雲端音視頻處理流程。

   功能強大：支援截圖、轉碼、窄帶高清分析、轉封裝、浮水印、剪輯等功能。

   詳細的控制台配置參考 ######。

2. 上傳多媒體檔案。

   把多媒體檔案上傳到媒體工作流程指定的路徑下，會自動觸發一個媒體工作流程執行，並按照指定的流程進行處理。

3. 等待訊息通知。

   執行開始、執行結束都會發送相應的訊息通知。

4. 播放。

   成功完成執行後，可以使用URL或者MediaId兩種方式來進行播放。

3 上傳視頻檔案

3.1 簡介

上傳

您可以使用上傳SDK上傳，支援Web（JavaScript）、移動（Android、iOS）

您也可以通過控制台和第三方工具上傳。

功能

簡單易用的API，只需指定本地檔案和OSS儲存位置，

斷點續傳、多檔案隊列、超大檔案、網路例外狀況事件、安全機制，

自動觸發媒體工作流程。

觸發媒體工作流程

多媒體檔案成功上傳到媒體工作流程指定的輸入bucket和路徑後，會自動觸發媒體工作流程的執行，按照媒體工作流程指定的流程進行處理。

使用上傳OSS檔案來自動觸發媒體工作流程的機制，需要滿足幾個條件：

- 匹配媒體工作流程

  工作流程觸發匹配規則，參見 #### ，並且工作流程的轉態是啟用

- 匹配副檔名

  觸發要求必須是多媒體檔案，媒體庫服務是通過副檔名來判斷的。檔案或者沒有副檔名(檔案名稱中不包含副檔名分割符號”.”)，或者副檔名符合下面的規則：

  - 視頻

    3gp、asf、avi、dat、dv、flv、f4v、gif、m2t、m3u8、m4v、mj2、mjpeg、mkv、mov、mp4、mpe、mpg、mpeg、mts、ogg、qt、rm、rmvb、swf、ts、vob、wmv、webm

  - 音頻

    aac、ac3、acm、amr、ape、caf、flac、m4a、mp3、ra、wav、wma、aiff

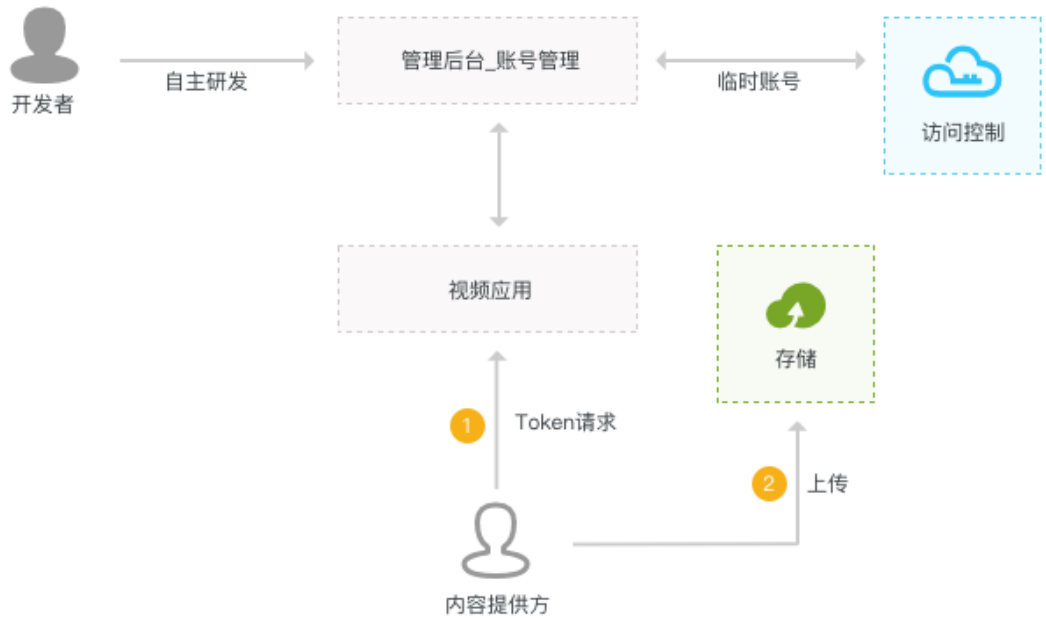
- 指定媒體屬性

  觸發時，也可以指定媒體屬性，包括：標題、標籤、描述、類目、封面URL、使用者自訂資料。詳細的屬性說明，參見 #### 文檔中的請求參數。

安全性

一般的使用情境是通過用戶端直接上傳視頻檔案。此時需要考慮AK在用戶端儲存的安全問題。泄露後，不僅風險大，而且不容易更換。所以建議用戶端訪問應用服務來擷取，並使用 ###### 提供的 #####。

建議流程



1. 請求安全性權杖。

   每次上傳前，使用者通過視頻應用（App或Web形式）訪問業務方的應用服務，應用服務從存取控制服務擷取安全性權杖(Token)，然後傳回給視頻應用。這樣即保證了安全性，還可以對使用者進行身分識別驗證和許可權控制，也可以記錄使用者上傳的曆史等。

   使用安全性權杖之前，需要先 ########。

   這裡提供 *Java#####*。（其他語言的使用方法參考 *STS##*）。

2. 上傳檔案。

   把上傳SDK整合到視頻應用後，可以通過擷取到的Token上傳檔案。參見 ####。

   - Web

     由於js檔案一般放在應用或CDN的網域名稱下，而視頻檔案放在OSS的網域名稱下，JavaScript上傳檔案時會涉及跨域請求，需要先 *##CORS*。

     JavaScript

- 移動

  Android

  iOS

3.2 設定子帳號和授權

請您按照以下步驟設定子帳號和授權。

1. 建立子帳號。

   a. 登入 *RAM ###*。

   b. 在左側導覽列中，單擊 使用者管理。

   c. 在 使用者管理 中，單擊 建立使用者。

   d. 在 建立使用者 中，輸入使用者資訊。

> 说明：
> 請您勾選 為該使用者自動產生**AccessKey**，並妥善儲存。後續需要子帳號的AccessKey擷取STS的Token。

2. 建立角色。

   a. 在左側導覽列中，單擊 角色管理。

   b. 單擊 建立角色。

   c. 在 選擇角色類型 中，選擇 使用者角色。

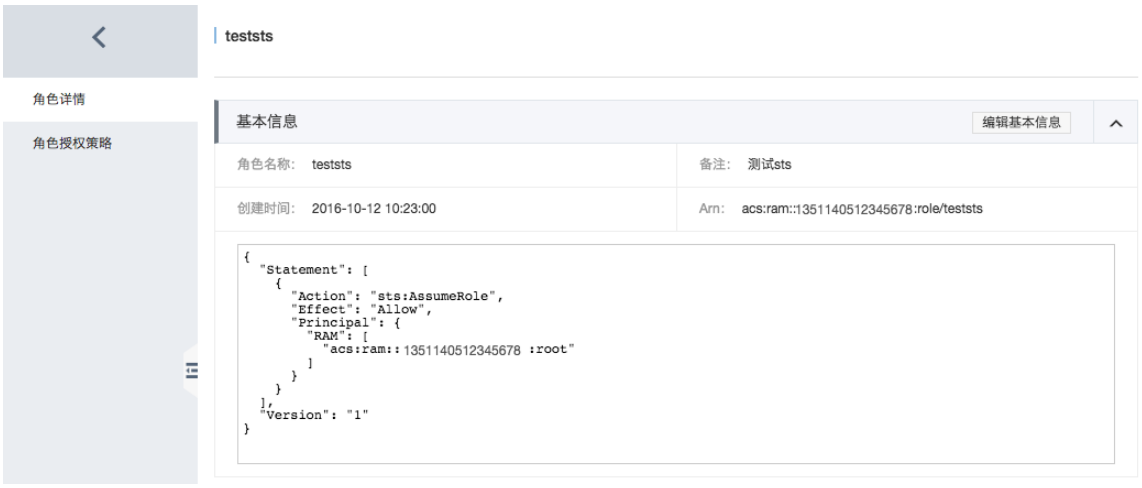
   d. 在 填寫類型資訊 中，選擇 當前雲帳號。

> 说明：
> 受信雲帳號**ID** 預設已經填寫了您當前雲帳號ID，無需修改，單擊 下一步 即可。

e. 在 配置角色基本資料 中，填寫 角色名稱，並單擊 建立。

f. 在 角色管理 中，單擊新建立的角色。

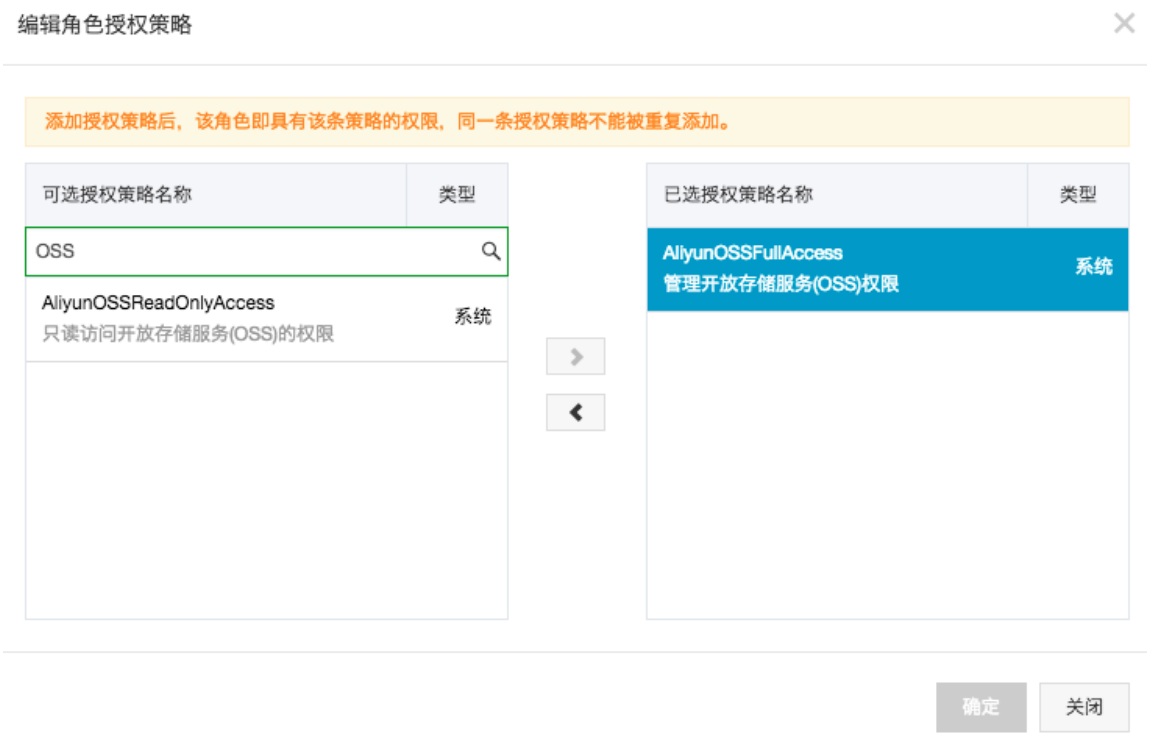
g. 在 角色詳情 中，複製 Arn 參數 acs:ram::1351140512345678:role/teststs。



3. 設定角色的許可權。

a. 單擊 角色授權策略。

b. 單擊 編輯授權策略。



说明：

如果您想要調整子帳號的STS許可權（例如，修改、增加、刪除等），只需回到本步驟操作即可。

上傳SDK需要的最小許可權可以在 ###### 中，建立一個授權策略來實現，然後在編輯角色授權策略中添加這個自訂授權策略即可。完整的策略內容如下：

```
{
     "Statement": [
       {
         "Action": [
           "oss:PutObject",
           "oss:AbortMultipartUpload",
           "oss:ListMultipartUploads",
           "oss:ListParts"
         ],
         "Effect": "Allow",
         "Resource": [
           "*"
         ]
       }
     ],
     "Version": "1"
   }
```

4. 關聯子帳號和角色。

a. 在 RAM控制台左側導覽列中，單擊 策略管理。

b. 選擇 自訂授權策略，並單擊 建立授權策略。

c. 在 選擇權限原則模板 > 全部模板 右側，輸入 **STS**，找到`AliyunSTSAssumeRoleAccess`的模板並單擊進入下一步。

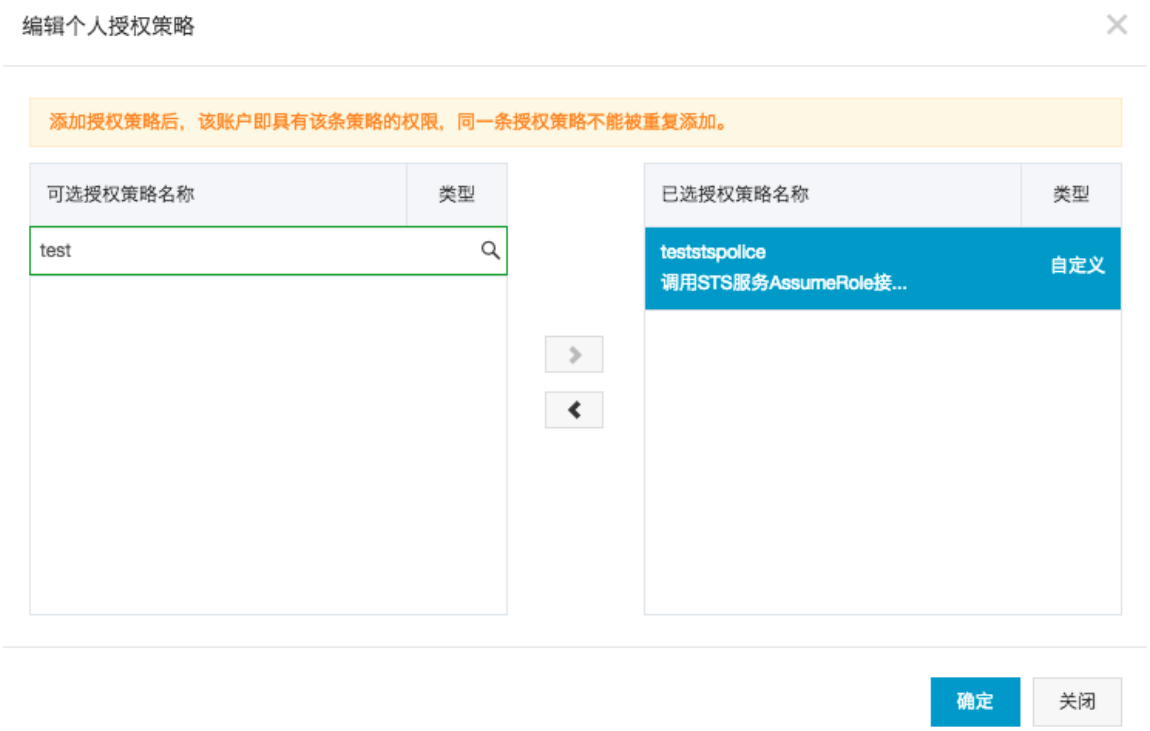
d. 在 編輯許可權並提交 中，輸入 授權策略名稱。

e. 在 策略內容 中，將 **Resource** 欄位，修改成您所擷取的 **Arn** 參數 **acs:ram::1351140512345678:role/teststs**。

f. 在RAM控制台的左側導覽列中，單擊 使用者管理。

g. 選擇您所設定的子帳號，並單擊 授權。

h. 輸入建立的策略test可以看到前面建立的`teststspolice`。

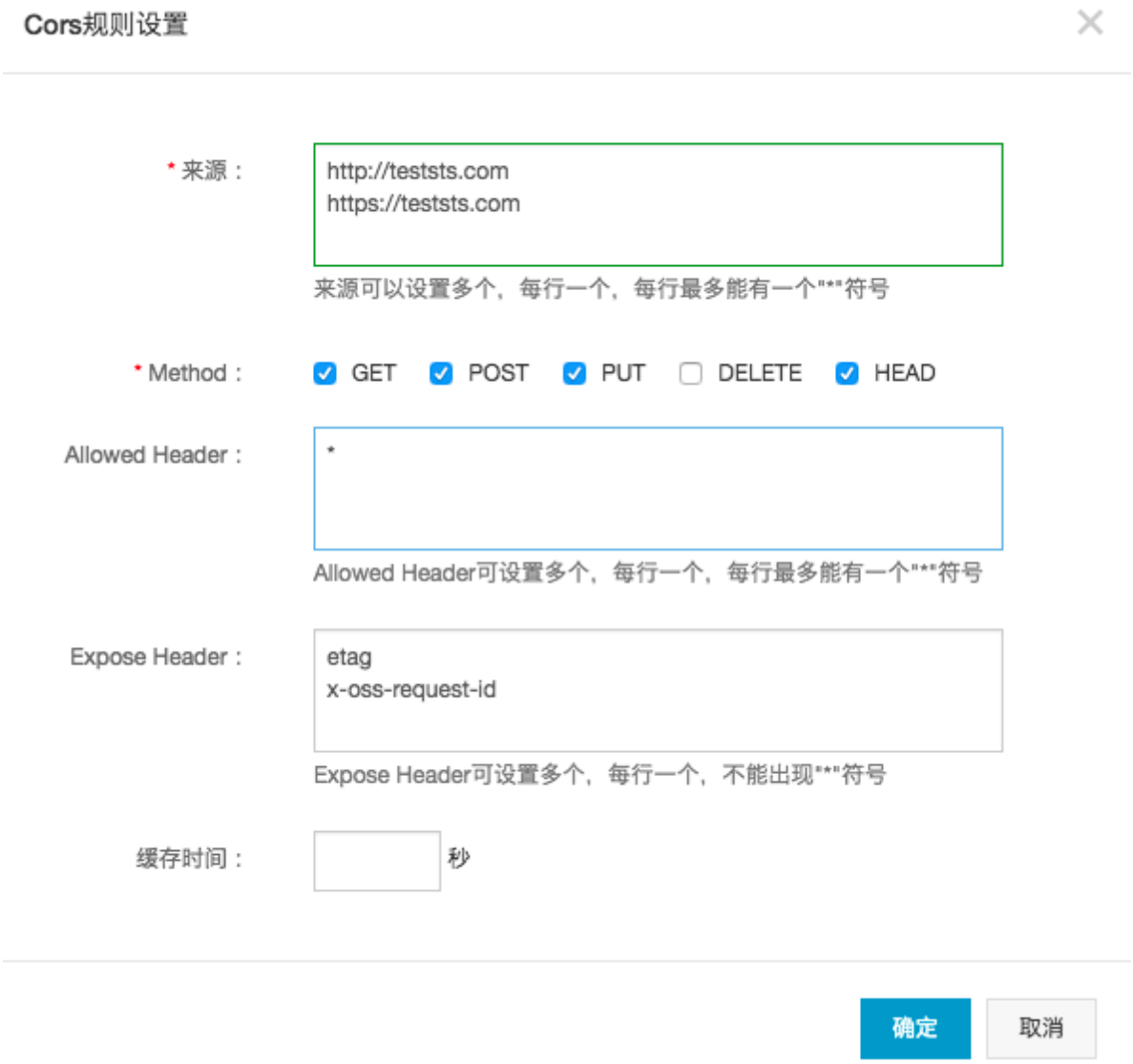


3.3 設定CORS

由於js檔案和視頻檔案放在不同的網域名稱下，JavaScript上傳檔案時會發起跨域請求。需要設定OSS的跨網域設定，否則無法通過Web端的JavaScript正常上傳檔案。

1. 開啟輸入bucket對應的CORS設定頁面。

2. 添加規則。



- 來源

  填寫實際部署js檔案的網域名稱，如果需要同時支援http和https訪問，需要分別添加。
- **Method**

  勾選 **GET**、**POST**、**PUT**、**HEAD**。
- **Allowed Header**

  填寫*。
- **Expose Header**

  分兩行填寫 **etag** 和 **x-oss-request-id**。

3.4 請求安全性權杖-Java範例程式碼

1. pom.xml中引用STS的SDK。

```
<repositories>
     <repository>
         <id>sonatype-nexus-staging</id>
         <name>Sonatype Nexus Staging</name>
         <url>https://oss.sonatype.org/service/local/staging/deploy/
maven2/</url>
         <releases>
             <enabled>true</enabled>
         </releases>
         <snapshots>
             <enabled>true</enabled>
         </snapshots>
     </repository>
 </repositories>
<dependencies>
 <dependency>
   <groupId>com.aliyun</groupId>
   <artifactId>aliyun-java-sdk-sts</artifactId>
   <version>2.1.6</version>
 </dependency>
 <dependency>
   <groupId>com.aliyun</groupId>
   <artifactId>aliyun-java-sdk-core</artifactId>
   <version>2.2.0</version>
 </dependency>
</dependencies>
```

2. 代碼。

   STS需要一個角色的參數`roleArn`。登入 *RAM ###*，單擊 角色管理，再單擊具體 角色名稱後，在基本資料中有一個參數`Arn`，例如`1351140512345678:role/teststs`。

   - main函數

```
public static void main(String[] args) throws Exception {
   IClientProfile profile = DefaultProfile.getProfile(
                                          "cn-hangzhou",
                                          <accessKeyId>,
                                          <accessKeySecret>);
   DefaultAcsClient client = new DefaultAcsClient(profile);
   AssumeRoleResponse response = assumeRole(client, <roleArn>);
   AssumeRoleResponse.Credentials credentials = response.
getCredentials();
   System.out.println(credentials.getAccessKeyId() + "\n" +
                      credentials.getAccessKeySecret() + "\n" +
                      credentials.getSecurityToken() + "\n" +
                      credentials.getExpiration());
}
```

   - 產生臨時AK和Token的函數

```
private static AssumeRoleResponse assumeRole(
                                    DefaultAcsClient client,
                                    String roleArn)
                                    throws ClientException {
```

```
    final AssumeRoleRequest request = new AssumeRoleRequest();
    request.setVersion("2015-04-01");
    request.setMethod(MethodType.POST);
    request.setProtocol(ProtocolType.HTTPS);
    request.setDurationSeconds(900L);
    request.setRoleArn(roleArn);
    request.setRoleSessionName("test-token");
    return client.getAcsResponse(request);
}
```

3. Token有效期間。

   範例程式碼中產生的Token有效時間為900秒，可以根據實際需求調整（最小900秒，最大3,600秒）。

   在有效期間內，不需要反覆產生新的Token，可以複用已經產生的Token。您可以通過以下方式判斷何時需要重建token：

```
private static boolean isTimeExpire(String expiration) {
     Date nowDate = new Date();
     Date expireDate = javax.xml.bind.DatatypeConverter.parseDateT
ime(expiration).getTime();
     if (expireDate.getTime() <= nowDate.getTime()) {
         return true;
     } else {
         return false;
     }
}
```

4 接收訊息通知

4.1 簡介

訊息格式

媒體工作流程開始執行和完成執行時，會向Message Service指定的隊列或主題(通知)發送訊息。

- 格式定義

  訊息體是JSON格式，詳細的欄位名稱、類型、描述參考 #### 的“媒體工作流程訊息”部分。

  結構的層次定義如下：

  - 頂層

    是一個JSON對象。定義：

    {當前活動的基本屬性、工作流程執行對象}

  - 當前活動的基本屬性

    當前活動的基本屬性不是一個獨立的對象，是直接屬於頂層的索引值屬性，可以參考下面的樣本。定義：

    工作流程執行ID、活動名字、活動類型、活動狀態、錯誤資訊。

  - 工作流程執行詳情對象

    是一個JSON對象。定義：

    {工作流程執行ID、媒體工作流程ID、媒體工作流程名字、媒體ID、輸入檔案、工作流程執行類型、使用中的物件數組、建立時間}

  - 使用中的物件數組

    是一個JSON數組，包含執行到目前狀態的所有活動。例如，開始訊息中只有一個Start使用中的物件，完成訊息則包含所有使用中的物件。定義：

    [使用中的物件、使用中的物件…]

  - 使用中的物件

    是一個JSON對象。定義：

    {活動名字、活動類型、作業ID、活動狀態、開始時間、結束時間、錯誤資訊}

- 開始

  活動基本屬性中“活動類型”是“Start”。

- 完成

  活動基本屬性中“活動類型”是“Report”。

- 樣本

```
{
      "RunId": "8f8aba5a62ab4127ae2add18da20b0f2",
      "Name": "Act-4",
      "Type": "Report",
      "State": "Success",
      "MediaWorkflowExecution": {
        "Name": "ConcurrentSuccess",
        "RunId": "8f8aba5a62ab4127ae2add18da20b0f2",
        "Input": {
            "InputFile": {
                "Bucket": "inputfirst",
                "Location": "oss-test",
                "Object": "mediaWorkflow/ConcurrentSuccess/01.wmv"
            }
        },
        "State": "Success",
        "MediaId": "2be491ab4cb6499cd0befe5fcf0cb670",
        "ActivityList": [
            {
                "RunId": "8f8aba5a62ab4127ae2add18da20b0f2",
                "Name": "Act-1",
                "Type": "Start",
                "State": "Success",
                "StartTime": "2016-03-15T02: 53: 41Z",
                "EndTime": "2016-03-15T02: 53: 41Z"
            },
            {
                "RunId": "8f8aba5a62ab4127ae2add18da20b0f2",
                "Name": "Act-2",
                "Type": "Transcode",
                "JobId": "f34b6d1429dd491faa7a6c1c8f905285",
                "State": "Success",
                "StartTime": "2016-03-15T02: 53: 43Z",
                "EndTime": "2016-03-15T02: 53: 47Z"
            },
            {
                "RunId": "8f8aba5a62ab4127ae2add18da20b0f2",
                "Name": "Act-3",
                "Type": "Snapshot",
                "JobId": "c14150be33304825a5d67cd5364c35cb",
                "State": "Success",
                "StartTime": "2016-03-15T02: 53: 44Z",
                "EndTime": "2016-03-15T02: 53: 45Z"
            },
            {
                "RunId": "8f8aba5a62ab4127ae2add18da20b0f2",
                "Name": "Act-4",
                "Type": "Report",
                "State": "Success",
                "StartTime": "2016-03-15T02: 53: 49Z",
                "EndTime": "2016-03-15T02: 53: 49Z"
            }
        ],
        "CreationTime": "2016-03-15T02: 53: 39Z"
      }
```

```
    }
```

接收和解析訊息

- 隊列

  PHP#####

- 主題（通知）

  PHP#####

4.2 隊列方式接收通知

本文介紹Message Service的要求和安裝說明。

詳情參見 *SDK##* 和 *######*。

樣本的語言採用PHP，其他語言使用說明，參見 *SDK####*。

環境要求

PHP 5.5+

安裝

從Aliyun下載Message Service的 *PHP SDK*

解壓到項目目錄，解壓後的目錄名是`php_sdk`。

範例程式碼

- 引用Message ServiceSDK

```
require_once(dirname(__FILE__).'/php_sdk/mns-autoloader.php');
```

- 初始化Message Service

  MNS對使用者的每個地區都配置了一個單獨的服務網域名稱，規則是：`https://${UserId}.mns.${Region}.aliyuncs.com`。範例程式碼使用的 華東**1**地區**(cn-hangzhou)**，也可以替換為其他地區，例如：華北**2**地區（cn-beijing）。

```
use AliyunMNS\Client;
 use AliyunMNS\Exception\MnsException;
```

```
$mns_client = new Client('https://'.$user_id.'.mns.cn-hangzhou.
aliyuncs.com',
                         $access_key_id, $access_key_secret);
 $queue = $mns_client->getQueueRef($queue_name);
```

- 接收訊息

MNS接收到的每條訊息都對應一個控制代碼，後續可以使用控制代碼操作這條訊息（例如：刪除訊息）。

另外，MNS支援批量接收訊息來提高效能。詳情參見MNS文檔 ######。

接收訊息時，可以指定逾時時間（樣本設定了3秒逾時），如果隊列中沒有訊息，會發生逾時並觸發異常。

```
$receipt_handle = NULL;
 $message = null;
 try
 {
      $res = $queue->receiveMessage(3);
      echo "ReceiveMessage Succeed! \n";
      $message = $res->getMessageBody();
      $receipt_handle = $res->getreceiptHandle();
 }
 catch (MnsException $e)
 {
      echo "ReceiveMessage Failed: " . $e . "\n";
 }
```

- 刪除訊息

  訊息不會主動從隊列刪除，必須主動調用刪除訊息，否則訊息會一直保持在隊列中，下次還會繼續接收到同一個訊息。另外，刪除訊息作業必須在接收到訊息後指定時間內調用才能成功，詳情參見 ####-######。

```
try
 {
      $res = $queue->deleteMessage($receipt_handle);
      echo "DeleteMessage Succeed! \n";
 }
 catch (MnsException $e)
 {
      echo "DeleteMessage Failed: " . $e . "\n";
 }
```

- 分析訊息

  訊息體是字串，內容是一個JSON對象，需要通過`json_decode`轉換成對象，然後就可以分析JSON對象來擷取詳細資料了，樣本列印了這次訊息是哪個輸出檔案觸發的媒體工作流程執行。

```
$json_message = json_decode($message);
 $input_file = $json_message->{'MediaWorkflowExecution'}->{'Input'}-
>{'InputFile'};
 echo '輸入檔案 location:'.$input_file->{'Location'}.
      ' bucket:'.$input_file->{'Bucket'}.
      ' object:'.$input_file->{'Object'}."\n";
```

- 擷取視頻輸出的詳細資料。

擷取到訊息詳細內容後，可以配合使用媒體庫服務API擷取工作流程執行的視頻詳細資料。樣本列印出這次轉碼和截圖作業的輸出地址。

如何安裝和配置媒體庫服務的PHP SDK，參考文檔 *###SDK-PHP*。

```
include_once 'aliyun-php-sdk-core/Config.php';
 use Mts\Request\V20140618 as Mts;
```

初始媒體庫服務的client。

```
$profile = DefaultProfile::getProfile('cn-hangzhou',
                                      $access_key_id,
                                      $access_key_secret);
 $mts_client = new DefaultAcsClient($profile);
```

列印所有轉碼作業的輸出地址和基本資料。

```
if (strcmp($json_message->{'Type'}, 'Report') == 0) {
      $activities = $json_message->{'MediaWorkflowExecution'}->{'
ActivityList'};
      $transcode_job_ids = Array();
      for ($i=0; $i < count($activities); $i++) {
        if (strcmp($activities[$i]->{'Type'}, 'Transcode') == 0) {
          $transcode_job_ids[] = $activities[$i]->{'JobId'};
        }
      }
      $request = new Mts\QueryJobListRequest();
      $request->setJobIds(join(',', $transcode_job_ids));
      $request->setRegionId('cn-hangzhou');
      $response = $mts_client->getAcsResponse($request);
      for ($i=0; $i < count($response->{'JobList'}->{'Job'}); $i++)
 {
        $output = $response->{'JobList'}->{'Job'}[$i]->{'Output'};
        $output_file = $response->{'JobList'}->{'Job'}[$i]->{'Output
'}->{'OutputFile'};
        $video_properties = $response->{'JobList'}->{'Job'}[$i]->{'
Output'}->{'Properties'};
        echo '轉碼輸出檔案URL '.'http://'.$output_file->{'Bucket'}.'.'.
                        $output_file->{'Location'}.'.aliyuncs.com/'.
                        urldecode($output_file->{'Object'})."\n";
        echo '轉碼輸出檔案基本資料 '.$video_properties->{'Width'}.'x'.$
video_properties->{'Height'}.
                        ' duration:'.$video_properties->{'Duration
'}."\n";
      }
 }
```

列印所有截圖作業的輸出地址。

```
if (strcmp($json_message->{'Type'}, 'Report') == 0) {
      $activities = $json_message->{'MediaWorkflowExecution'}->{'
ActivityList'};
      $snapshot_job_ids = Array();
      for ($i=0; $i < count($activities); $i++) {
        if (strcmp($activities[$i]->{'Type'}, 'Snapshot') == 0) {
          $snapshot_job_ids[] = $activities[$i]->{'JobId'};
        }
```

```
        }
        $request = new Mts\QuerySnapshotJobListRequest();
        $request->setSnapshotJobIds(join(',', $snapshot_job_ids));
        $request->setRegionId('cn-hangzhou');
        $response = $mts_client->getAcsResponse($request);
        for ($i=0; $i < count($response->{'SnapshotJobList'}->{'
SnapshotJob'}); $i++) {
          $snapshot_config = $response->{'SnapshotJobList'}->{'
SnapshotJob'}[$i]->{'SnapshotConfig'};
          $output_file = $response->{'SnapshotJobList'}->{'SnapshotJob
'}[$i]->{'SnapshotConfig'}->{'OutputFile'};
          echo '截圖輸出檔案URL '.'http://'.$output_file->{'Bucket'}.'.'.
                          $output_file->{'Location'}.'.aliyuncs.com/'.
                          urldecode($output_file->{'Object'})."\n";
        }
 }
```

4.3 主題通知方式接收訊息

主題（通知）是Message Service主動推送訊息給使用者，只要有能公開訪問的HTTP服務即可接收。

基本結構

視頻媒體庫通知的基本結構是這樣的：

- 最外層是Message Service的結構體

  Message Service的詳細定義和格式，詳情參考 *Message Service-Notification##*。

- Message Service的訊息本文是媒體庫服務的結構體

  取到訊息本文後，就能進一步對媒體庫服務的訊息進行解析了，詳情參考 ######### ，文檔中有詳細的解析步驟和範例程式碼。

安全

主題（通知）方式是公開訪問的HTTP服務，所以要避免非法的攻擊調用。關於識別訊息是否由Message Service發起，請您參見Message Service文檔 *Endpoint####*。

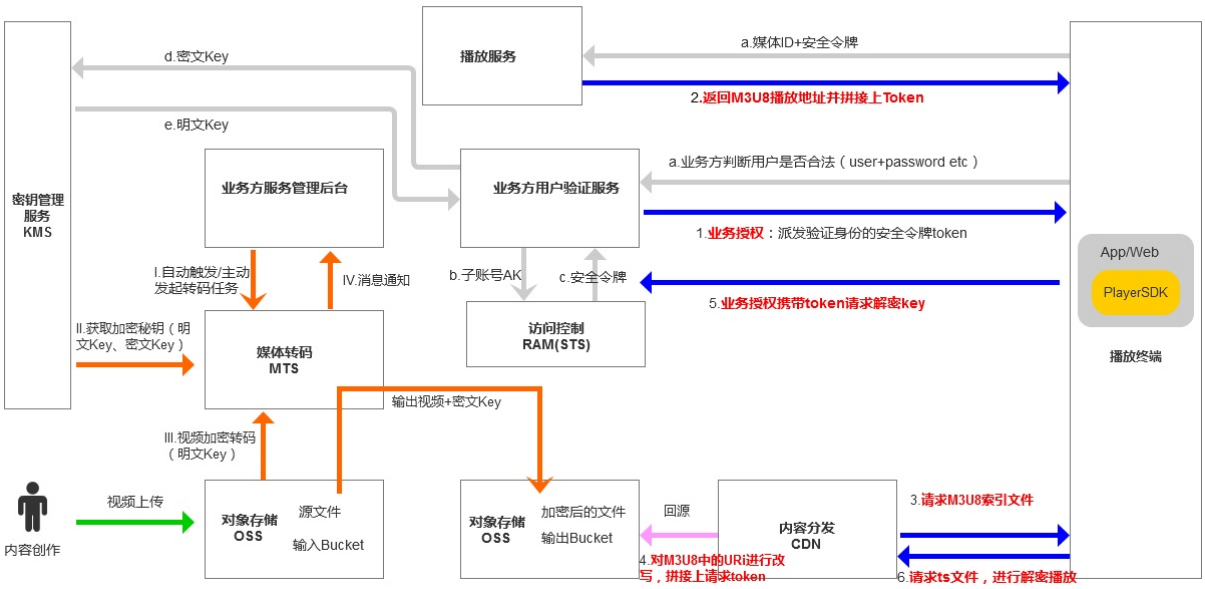
5 視頻加密

5.1 HLS標準加密

視頻加密是對視頻內容保護的一種手段，對視頻中的內容進行加密，可有效防止視頻泄露和盜鏈問題，廣泛用於線上教育及財經等領域。

> 说明：
> 阿里雲目前支援兩種加密方式: 私人加密 和 HLS標準加密，HLS標準加密需要客戶自己保護密鑰，此文檔介紹HLS標準加密。

加密架構



術語介紹

- 秘鑰管理服務（Key Management Service，簡稱KMS）

  一項安全管理服務，主要負責資料秘鑰的生產、加密、解密等工作。開通請 ####。

- 資料秘鑰（Data Key，簡稱DK）也稱清除金鑰

  DK為加密資料使用的明文資料密鑰。

- 信封資料密鑰（Enveloped Data Key，簡稱EDK）也稱密文密鑰

  EDK為通過信封加密技術保密後的密文資料密鑰。

- 存取控制（Resource Access Management 簡稱RAM）

  是阿里雲為客戶提供的 使用者身份管理 與 資源存取控制 服務。開通請 ####。

操作流程

1. 建立HLS加密工作流程。

   > 说明：
   > 控制台暫不支援建立HLS加密工作流程，請通過API進行建立。查看demo，參見 *##HLS########*。建立好後，不要在控制台對其修改，會使加密配置失效。

   工作流程中關鍵配置：

   - 開始活動節點：`InputFile:{"Bucket":"bucketdemo", "Location ":"oss-cn-hangzhou", "ObjectPrefix":"HLS-Encryption"}`

     此配置表示內容創作者上傳視頻到杭州 oss://bucketdemo/HLS-Encryption 這個路徑下會自動觸發加密轉碼。
   - 轉碼活動節點：`Encryption:{"Type":"hls-aes-128", "KeyUri":"https://decrypt.demo.com"}`

     轉碼完成後，KeyUri的配置會出現在m3u8檔案中，供播放器使用。

     播放時播放器會攜帶EDK密文密鑰請求這個地址，以擷取DK清除金鑰，進行播放。
2. 上傳視頻。

   兩種方法上傳視頻，都會自動觸發加密轉碼。

   - 通過MPS控制台上傳視頻至剛剛建立的工作流程。
   - 通過OSS上傳工具上傳視頻至oss://bucketdemo/HLS-Encryption路徑。

   轉碼完成後，m3u8檔案內容樣本。

   ```
   #EXTM3U
        #EXT-X-VERSION:3
        #EXT-X-TARGETDURATION:5
        #EXT-X-MEDIA-SEQUENCE:0
        #EXT-X-KEY:METHOD=AES-128,URI="https://decrypt.demo.com?
   Ciphertext=aabbccddeeff&MediaId=fbbf98691ea44b7c82dd75c5bc8b9271"
        #EXTINF:4.127544,
        15029611683170-00001.ts
        #EXT-X-ENDLIST
   ```
3. 播放。

   - 擷取播放地址，可通過查詢媒體介面，參見 *####-####ID* 擷取OSS地址，然後把OSS網域名稱，替換為CDN網域名稱，再拼接上參數**`MtsHlsUriToken`**，此參數會作為請求解密密鑰的令牌，以下為原理。

播放時，播放器會訪問m3u8檔案中EXT-X-KEY標籤中的URI以擷取解密秘鑰，此URI為業務方搭建的解密密鑰介面，只允許合法使用者訪問。 因此，需要播放器在請求解密時，攜帶一些業務方承認的認證資訊。MtsHlsUriToken就是這個作用，業務方頒發一令牌給播放器， 播放器請求解密密鑰時，攜帶此令牌，業務方驗證令牌的合法性。

- 播放器攜帶令牌，業務方驗證服務。

  例如，正常的播放地址為：`https://vod.demo.com/test.m3u8`, 當拼接攜帶`MtsHlsUriToken`參數後為`https://vod.demo.com/test.m3u8?MtsHlsUriToken=`業務方頒發的令牌。

  播放時，播放器向阿里CDN請求`https://vod.demo.com/test.m3u8?MtsHlsUriToken=`業務方頒發的令牌，阿里CDN會動態修改m3u8檔案中的解密URI，如原為`https://decrypt.demo.com?Ciphertext=aabbccddeeff&MediaId=fbbf98691ea44b7c82dd75c5bc8b9271`，修改後為 `https://decrypt.demo.com?Ciphertext=aabbccddeeff&MediaId=fbbf98691ea44b7c82dd75c5bc8b9271&MtsHlsUriToken=`業務方頒發的令牌。

  所以，播放器最終請求解密URI為：`https://decrypt.demo.com?Ciphertext=aabbccddeeff&MediaId=fbbf98691ea44b7c82dd75c5bc8b9271&MtsHlsUriToken=`業務方頒發的令牌。此地址中，攜帶了業務方搬發的令牌，業務方進行驗證即可。

4. 業務方需要完成以下操作。

   a. 搭建頒發及驗證MtsHlsUriToken令牌服務。

   b. 校正解密令牌，推薦一個令牌只允許使用一次。

   c. 解密密鑰：EDK即Ciphertext，此時，要調用KMS服務的解密介面進行解密，參見 *Decrypt*。解密後，可緩衝，以減少不必要的網路IO。

   d. 解密拿到DK即清除金鑰，需要base64decode，然後返回給播放器。

6 媒體庫管理

6.1 簡介

您可以使用媒體庫SDK，支援 *Java*、*PHP*、*Python*。

也可以直接通過HTTP、HTTPS訪問API，詳情參考 *API####*。

功能

媒體工作流程管理：增、刪、改、查以及啟用和停止。

媒體工作流程執行執行個體管理：遍曆和查詢。

媒體管理：增、刪、改、查和搜尋。以及媒體屬性的維護（標題、標籤、封面、描述）。以及媒體發布狀態的設定。

媒體類目管理：增、刪、改、查。

業務情境

- 搜尋媒體

  在媒體庫中搜尋滿足條件的媒體集合。

  可以通過關鍵字來搜尋，是邏輯“或”的關係：只要標題、標籤、描述、類目任一屬效能匹配即可。也可以通過組合條件來搜尋，是邏輯“與”的關係：指定的組合屬性中（標題、標籤、描述、類目），必須每個屬性都匹配。

  在搜尋條件中，可以指定建立時間的區間來限定搜尋範圍。返回的結果也可以指定定序：按照建立時間升序或降序。

  另外，返回的介面比較多時，可以選擇分頁擷取。
- 維護媒體屬性

  每個媒體都有基本屬性：標題、標籤、描述、類目。都可以通過API來設定。

  ####-#####-PHP
- 管理媒體標籤

  媒體庫不提供全域的標籤管理，每個媒體的標籤都是獨立的設定，可以通過搜尋媒體的API來尋找所有設定了相同標籤的媒體。

  ####-#####-PHP

管理媒體類目

媒體庫提供了一個全域的類目管理，每個媒體可以關聯到一個類目，並結合搜尋媒體功能來快速檢索。

####-#####-PHP

查詢媒體詳細資料

一個媒體包含一個輸入和若干個輸出（視頻、截圖等）。可以通過查詢返回媒體的詳細輸入、輸出的資訊。

輸入資訊包含：視頻基本屬性（寬、高、時間長度、大小、碼率、幀率）以及視頻詳情（容器封裝、視頻、音頻、字幕流，以及封裝和流的詳細屬性）。

輸出資訊包含：視頻有基本屬性（寬、高、時間長度、大小、碼率、幀率）以及OSS的URL地址。截圖有類型（單幀、批量）以及OSS的URL地址。

######-#####-PHP

6.2 視頻基本屬性

簡介

本文介紹如何查詢和更新媒體基本資料。SDK的安裝和使用，參見 *###SDK-PHP*。

查詢媒體基本資料

查詢媒體提供了2種方式：媒體ID或OSS檔案地址。

- 使用媒體ID查詢媒體

  詳細參數參考 *API#### > #### > #### > ####ID*。

```
include_once 'aliyun-php-sdk-core/Config.php';
use Mts\Request\V20140618 as Mts;
$accessKeyID = 'test'; // 替換成真實的id
$accessKeySecret = 'test'; // 替換成真實的secret
$profile = DefaultProfile::getProfile('cn-hangzhou',
                                      $accessKeyID,
                                      $accessKeySecret);
$client = new DefaultAcsClient($profile);
```

```
function queryMediaById($client, $mediaID)
{
     $request = new Mts\QueryMediaListRequest();
     $request->setAcceptFormat('JSON');
     $request->setMediaIds($mediaID);
     $response = $client->getAcsResponse($request);
     return $response;
}
function printMedia($media)
```

```
{
     if (array_key_exists('Title', $media)) {
       print_r('Title: '.$media->{'Title'}."\n");
     }
     if (array_key_exists('Description', $media)) {
       print_r('Description: '.$media->{'Description'}."\n");
     }
     if (array_key_exists('Tags', $media)) {
       print_r('Tags: '.$media->{'Tags'}->{'Tag'}[0]."\n");
     }
     if (array_key_exists('CoverURL', $media)) {
       print_r('CoverURL: '.$media->{'CoverURL'}."\n");
     }
     print_r('Format: '.$media->{'Format'}."\n");
     print_r('Resolution: '.$media->{'Width'}.'x'.$media->{'Height
'}."\n");
     print_r('FileSize: '.$media->{'Size'}."\n");
     print_r('Bitrate: '.$media->{'Bitrate'}."\n");
     print_r('FPS: '.$media->{'Fps'}."\n");
}
$mediaID = 'test'; // 替換成真實的mediaID
$medias = queryMediaById($client, $mediaID)->{'MediaList'}->{'Media
'};
for ($i=0; $i < count($medias); $i++) {
     printMedia($medias[$i]);
}
```

- 使用OSS檔案地址查詢媒體

  詳細參數參考 *API#### > #### > #### > ##OSS####*。

```
function queryMediaByURL($client, $mediaURL)
{
     $request = new Mts\QueryMediaListByURLRequest();
     $request->setAcceptFormat('JSON');
     $request->setFileURLs($mediaURL);
     $response = $client->getAcsResponse($request);
     return $response;
}
$ossEndpoint = 'http://test.oss-cn-hangzhou.aliyuncs.com/';
// OSS的Object不需要"/"開始，替換成真實的ossObject
$ossObject = 'test/測試.mp4';
$medias = queryMediaByURL($client,$ossEndpoint.urlencode($ossObject
))->{'MediaList'}->{'Media'};
for ($i=0; $i < count($medias); $i++) {
     printMedia($medias[$i]);
}
```

- 更新屬性

  更新提供了2種更新方式：全量屬性更新，單個屬性更新。

  ▬ 全量更新屬性

  詳細參數參考 *API#### > #### > #### > ####*。

更新時，必須指定所有欄位，不設定的欄位會被清空。

```
function updateMediaAllField($client, $mediaID, $title, $
description, $tags, $coverURL)
{
     $request = new Mts\UpdateMediaRequest();
     $request->setAcceptFormat('JSON');
     $request->setMediaId($mediaID);
     $request->setTitle($title);
     $request->setCateId(2663987);
     $request->setDescription($description);
     $request->setTags($tags);
     $request->setCoverURL($coverURL);
     $response = $client->getAcsResponse($request);
     return $response;
}
$mediaID = 'test'; // 替換成真實的mediaID
$media = updateMediaAllField($client, $mediaID,
                        'title', 'description', 'tags', 'coverURL
')->{'Media'};
```

- 單個更新屬性

不同的欄位可以單獨更新，使用的是不同API，可以不修改其他欄位的情況下，方便的更新單個欄位。

這裡通過“發布狀態”舉例，詳細參數參考 *API#### > #### > #### > ####*。

```
function updateMediaPublishState($client, $mediaID, $state)
{
     $request = new Mts\UpdateMediaPublishStateRequest();
     $request->setAcceptFormat('JSON');
     $request->setMediaId($mediaID);
     $request->setPublish($state);
     $response = $client->getAcsResponse($request);
     return $response;
}
$mediaID = 'test'; // 替換成真實的mediaID
// 更新"發布狀態"的API沒有傳回值，通過捕獲異常來判斷是否執行成功
try {
     updateMediaPublishState($client, $mediaID, "true");
} catch (ClientException $e) {
     print_r('ClientException:'."\n");
     print_r($e);
} catch (ServerException $e) {
     print_r('ServerException:'."\n");
     print_r($e);
}
```

6.3 媒體詳細資料

簡介

SDK的安裝和使用，詳情參考 *###SDK-PHP*

一個媒體包含一個輸入檔案和若干輸出檔案。輸入除了基本資料之外，還有詳細的 ####。輸出可以查詢視頻和截圖的詳細資料。

輸入媒體資訊

```
include_once 'aliyun-php-sdk-core/Config.php';
 use Mts\Request\V20140618 as Mts;
 $accessKeyID = 'test'; // 替換成真實的id
 $accessKeySecret = 'test'; // 替換成真實的secret
 $profile = DefaultProfile::getProfile('cn-hangzhou',
                                       $accessKeyID,
                                       $accessKeySecret);
 $client = new DefaultAcsClient($profile);
```

```
function queryMedia($client, $mediaID)
{
    $request = new Mts\QueryMediaListRequest();
    $request->setAcceptFormat('JSON');
    $request->setMediaIds($mediaID);
    $request->setIncludeMediaInfo("true");
    $response = $client->getAcsResponse($request);
    return $response;
}
function printMediaInfo($mediaInfo)
{
    print_r('Number of Streams: '.$mediaInfo->{'Format'}->{'NumStreams
'}."\n");
    if (array_key_exists('Streams', $mediaInfo) &&
        array_key_exists('AudioStreamList', $mediaInfo->{'Streams'})
 &&
        array_key_exists('AudioStream', $mediaInfo->{'Streams'}->{'
AudioStreamList'})) {
        $audioStreams = $mediaInfo->{'Streams'}->{'AudioStreamList'}-
>{'AudioStream'};
        print_r('Audio Streams:'."\n");
        for ($i = 0; $i < count($audioStreams); $i++) {
            print_r("\t[".$i."]"."\n");
            print_r("\t\tCodecName: ".$audioStreams[$i]->{'CodecName
'}."\n");
            print_r("\t\tChannels: ".$audioStreams[$i]->{'Channels
'}."\n");
            print_r("\t\tSamplerate: ".$audioStreams[$i]->{'Samplerate
'}."\n");
            print_r("\t\tDuration: ".$audioStreams[$i]->{'Duration
'}."\n");
            print_r("\t\tBitrate: ".$audioStreams[$i]->{'Bitrate'}."\n
");
        }
    }
    if (array_key_exists('Streams', $mediaInfo) &&
        array_key_exists('VideoStreamList', $mediaInfo->{'Streams'})
 &&
        array_key_exists('VideoStream', $mediaInfo->{'Streams'}->{'
VideoStreamList'})) {
        $videoStreams = $mediaInfo->{'Streams'}->{'VideoStreamList'}-
>{'VideoStream'};
        print_r('Video Streams:'."\n");
        for ($i = 0; $i < count($videoStreams); $i++) {
            print_r("\t[".$i."]"."\n");
```

```
            print_r("\t\tCodecName: ".$videoStreams[$i]->{'CodecName
'}."\n");
            print_r("\t\tProfile: ".$videoStreams[$i]->{'Profile'}."\n
");
            print_r("\t\tDuration: ".$videoStreams[$i]->{'Duration
'}."\n");
            print_r("\t\tPixFmt: ".$videoStreams[$i]->{'PixFmt'}."\n
");
            print_r("\t\tFps: ".$videoStreams[$i]->{'Fps'}."\n");
            print_r("\t\tBitrate: ".$videoStreams[$i]->{'Bitrate'}."\n
");
            print_r("\t\tResolution: ".$videoStreams[$i]->{'Width'}.'x
'.$videoStreams[$i]->{'Height'}."\n");
        }
    }
}
$mediaID = 'test'; // 替換成真實的mediaID
$medias = queryMedia($client, $mediaID)->{'MediaList'}->{'Media'};
for ($i = 0; $i < count($medias); $i++) {
    printMediaInfo($medias[$i]->{'MediaInfo'});
}
```

輸出

- **視頻**

```
function queryMedia($client, $mediaID)
{
    $request = new Mts\QueryMediaListRequest();
    $request->setAcceptFormat('JSON');
    $request->setMediaIds($mediaID);
    $request->setIncludePlayList("true");
    $response = $client->getAcsResponse($request);
    return $response;
}
function printOutputVideos($videos)
{
    print_r('Number of Output Video: '.count($videos)."\n");
    for ($i = 0; $i < count($videos); $i++) {
        print_r("\t[".$i."]"."\n");
        print_r("\t\tMediaWorkflowName: ".$videos[$i]->{'MediaWorkf
lowName'}."\n");
        print_r("\t\tActivityName: ".$videos[$i]->{'ActivityName
'}."\n");
        print_r("\t\tFormat: ".$videos[$i]->{'Format'}."\n");
        print_r("\t\tDuration: ".$videos[$i]->{'Duration'}."\n");
        print_r("\t\tFps: ".$videos[$i]->{'Fps'}."\n");
        print_r("\t\tBitrate: ".$videos[$i]->{'Bitrate'}."\n");
        print_r("\t\tSize: ".$videos[$i]->{'Size'}."\n");
        print_r("\t\tResolution: ".$videos[$i]->{'Width'}.'x'.$
videos[$i]->{'Height'}."\n");
        print_r("\t\tURL: ".$videos[$i]->{'File'}->{'URL'}."\n");
    }
}
$mediaID = 'test'; // 替換成真實的mediaID
$medias = queryMedia($client, $mediaID)->{'MediaList'}->{'Media'};
for ($i = 0; $i < count($medias); $i++) {
    printOutputVideos($medias[$i]->{'PlayList'}->{'Play'});
```

```
}
```

- 截圖

```
function queryMedia($client, $mediaID)
{
    $request = new Mts\QueryMediaListRequest();
    $request->setAcceptFormat('JSON');
    $request->setMediaIds($mediaID);
    $request->setIncludeSnapshotList("true");
    $response = $client->getAcsResponse($request);
    return $response;
}
function printOutputSnapshots($snapshots)
{
    print_r('Number of Output Snapshot: '.count($snapshots)."\n");
    for ($i = 0; $i < count($snapshots); $i++) {
        print_r("\t[".$i."]"."\n");
        print_r("\t\tMediaWorkflowName: ".$snapshots[$i]->{'
MediaWorkflowName'}."\n");
        print_r("\t\tActivityName: ".$snapshots[$i]->{'ActivityName
'}."\n");
        print_r("\t\tType: ".$snapshots[$i]->{'Type'}."\n");
        print_r("\t\tCount: ".$snapshots[$i]->{'Count'}."\n");
        print_r("\t\tURL: ".$snapshots[$i]->{'File'}->{'URL'}."\n");
    }
}
$mediaID = 'test'; // 替換成真實的mediaID
$medias = queryMedia($client, $mediaID)->{'MediaList'}->{'Media'};
for ($i = 0; $i < count($medias); $i++) {
    printOutputSnapshots($medias[$i]->{'SnapshotList'}->{'Snapshot
'});
}
```

6.4 標籤管理

簡介

SDK的安裝和使用，詳情參考 *###SDK-PHP*。

媒體庫不提供全域的標籤管理和設定，每個媒體的標籤都是獨立的。可以通過搜尋媒體的API來尋找所有設定了相同標籤的媒體。

標籤的API支援單個標籤的添加和刪除，如果要一次設定多個標籤，可以通過 *####-####* 實現。

添加標籤

詳細參數參考 *API#### > #### > #### > ####*。

```
include_once 'aliyun-php-sdk-core/Config.php';
  use Mts\Request\V20140618 as Mts;
  $accessKeyID = 'test'; // 替換成真實的id
  $accessKeySecret = 'test'; // 替換成真實的secret
  $profile = DefaultProfile::getProfile('cn-hangzhou',
                                        $accessKeyID,
                                        $accessKeySecret);
```

```
  $client = new DefaultAcsClient($profile);
```

```
function addMediaTag($client, $mediaID, $tag)
  {
      $request = new Mts\AddMediaTagRequest();
      $request->setAcceptFormat('JSON');
      $request->setMediaId($mediaID);
      $request->setTag($tag);
      $response = $client->getAcsResponse($request);
      return $response;
  }
  $mediaID = 'test'; // 替換成真實的mediaID
  // API沒有傳回值，通過捕獲異常來判斷是否執行成功
  try {
      addMediaTag($client, $mediaID, "testtag");
  } catch (ClientException $e) {
      print_r('ClientException:'."\n");
      print_r($e);
  } catch (ServerException $e) {
      print_r('ServerException:'."\n");
      print_r($e);
  }
```

刪除 標籤

詳細參數參考 *API#### > #### > #### > ####*。

```
function deleteMediaTag($client, $mediaID, $tag)
  {
      $request = new Mts\DeleteMediaTagRequest();
      $request->setAcceptFormat('JSON');
      $request->setMediaId($mediaID);
      $request->setTag($tag);
      $response = $client->getAcsResponse($request);
      return $response;
  }
  $mediaID = 'test'; // 替換成真實的mediaID
  // API沒有傳回值，通過捕獲異常來判斷是否執行成功
  try {
      deleteMediaTag($client, $mediaID, "testtag");
  } catch (ClientException $e) {
      print_r('ClientException:'."\n");
      print_r($e);
  } catch (ServerException $e) {
      print_r('ServerException:'."\n");
      print_r($e);
  }
```

6.5 類目管理

簡介

SDK的安裝和使用，參見 *###SDK-PHP*。

類目支援增刪改查。您需要關注以下內容：

- 刪除一個類目並不會自動清除關聯媒體的類目ID屬性。

- 查詢類目返回介面有兩種形式：樹結構、列表結構。樹結構返回的是一個嵌套結構的JSON對象，列表結構返回的是一個平面的數組結構，可以根據情境選擇使用。

新增類目

參見 *API#### > ###### > ####*。

```
include_once 'aliyun-php-sdk-core/Config.php';
  use Mts\Request\V20140618 as Mts;
  $accessKeyID = 'test'; // 替換成真實的id
  $accessKeySecret = 'test'; // 替換成真實的secret
  $profile = DefaultProfile::getProfile('cn-hangzhou',
                                        $accessKeyID,
                                        $accessKeySecret);
  $client = new DefaultAcsClient($profile);
```

```
function addCategory($client, $parentId, $categoryName)
  {
      $request = new Mts\AddCategoryRequest();
      $request->setAcceptFormat('JSON');
      $request->setParentId($parentId);
      $request->setCateName($categoryName);
      $response = $client->getAcsResponse($request);
      return $response;
  }
  $category = addCategory($client, null, 'testroot')->{'Category'};
  print_r('Level: '.$category->{'Level'}.
          "\tParentId: ".$category->{'ParentId'}.
          "\tCateId: ".$category->{'CateId'}.
          "\tCateName: ".$category->{'CateName'}."\n");
```

更新類目

參見 *API#### > ###### > ####*。

```
function updateCategory($client, $categoryId, $categoryName)
  {
      $request = new Mts\UpdateCategoryNameRequest();
      $request->setAcceptFormat('JSON');
      $request->setCateId($categoryId);
      $request->setCateName($categoryName);
      $response = $client->getAcsResponse($request);
      return $response;
  }
  try {
      updateCategory($client, 12345678, 'updatetestroot'); // 替換成真實
的CateID
  } catch (ClientException $e) {
      print_r('ClientException:'."\n");
      print_r($e);
  } catch (ServerException $e) {
      print_r('ServerException:'."\n");
      print_r($e);
```

```
  }
```

刪除類目

參見 *API#### > ###### > ####*。

```
function deleteCategory($client, $categoryId)
  {
      $request = new Mts\DeleteCategoryRequest();
      $request->setAcceptFormat('JSON');
      $request->setCateId($categoryId);
      $response = $client->getAcsResponse($request);
      return $response;
  }
  try {
      deleteCategory($client, 12345678); // 替換成真實的CateID
  } catch (ClientException $e) {
      print_r('ClientException:'."\n");
      print_r($e);
  } catch (ServerException $e) {
      print_r('ServerException:'."\n");
      print_r($e);
  }
```

查詢類目

- 樹結構

  參見 *API#### > ###### > #### > #*。

  ```
  function queryCategoryTree($client)
      {
          $request = new Mts\CategoryTreeRequest();
          $request->setAcceptFormat('JSON');
          $response = $client->getAcsResponse($request);
          return $response;
      }
      function printCategoryTree($categoryTree)
      {
          foreach($categoryTree as $category) {
              for ($i = 0; $i < $category->{'Level'}; $i++) {
                  print_r("--");
              }
              print_r('Level: '.$category->{'Level'}.
                      "\tParentId: ".$category->{'ParentId'}.
                      "\tCateId: ".$category->{'CateId'}.
                      "\tCateName: ".$category->{'CateName'}."\n");
              if (array_key_exists('SubcateList', $category)) {
                  printCategoryTree($category->{'SubcateList'});
              }
          }
      }
      $categoryTree = queryCategoryTree($client)->{'CategoryTree'};
      printCategoryTree(json_decode($categoryTree));
  ```

- 列表結構

參見 *API#### > ###### > #### > ##*。